

AL/HR-TP-1996-0013



**INTELLIGENT AGENTS
FOR COMPUTER-GENERATED FORCES**

**Gary R. George
Ellen Mallery
Marie Pope**

**Hughes Training, Inc.
Training Operations
6001 S. Power Road, Bldg 561
Mesa, AZ 85206-0904**

**HUMAN RESOURCES DIRECTORATE
AIRCREW TRAINING RESEARCH DIVISION
6001 S. Power Road, Bldg 558
Mesa, AZ 85206-0904**

August 1996

Final Technical Paper for Period September 1995 to December 1995

Approved for public release; distribution is unlimited.

**AIR FORCE MATERIEL COMMAND
BROOKS AIR FORCE BASE, TEXAS**

**ARMSTRONG
LABORATORY**

NOTICES

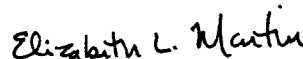
When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Office of Public Affairs has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This report has been reviewed and is approved for publication.



HERBERT H. BELL
Project Scientist



ELIZABETH L. MARTIN
Technical Director



LYNN A. CARROLL, Colonel, USAF
Chief, Aircrew Training Research Division

Please notify AL/HRPP, 7909 Lindbergh Drive, Brooks AFB, TX 78235-5352, if your address changes, or if you no longer want to receive our technical reports. You may write or call the STINFO Office at DSN 240-3877 or commercial (210) 536-3877.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE August 1996	3. REPORT TYPE AND DATES COVERED Final - Septemeber 1995 to December 1995	
4. TITLE AND SUBTITLE Intelligent Agents for Computer-Generated Forces			5. FUNDING NUMBERS C - F41624-95-C-5011 PE - 63227F PR - 2743 TA - 25 WU - 06	
6. AUTHOR(S) Gary R. George Ellen Mallery Marie Pope				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Hughes Training, Inc. Training Operations 6001 South Power Road, Building 561 Mesa, AZ 85206-0904			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Armstrong Laboratory Human Resources Directorate Aircrew Training Research Division 6001 South Power Road, Building 558 Mesa, AZ 85206-0904			10. SPONSORING/MONITORING AGENCY REPORT NUMBER AL/HR-TP-1996-0013	
11. SUPPLEMENTARY NOTES Armstrong Laboratory Technical Monitor: Dr Herbert H. Bell, (602) 988-6561				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) Computer-generated forces (CGF) are an important part of today's training devices. When linked to manned simulators, these computer-generated entities provide a dynamic and realistic environment for interaction of human participants. It also allows the addition of many players, which might not be cost effective using many networked, manned devices as an alternative. These CGFs are comprised of two specific objects: equipment model and a behavioral or cognitive model. The equipment model represents the machine which, in this case, is an aircraft with its associated dynamics, weapons systems, controls, and avionics systems. The cognitive model corresponds to how the machine operator, a pilot in this case, reacts in the dynamic environment. This will be based on mission knowledge, tactical doctrine, and situation awareness. Modeling of the cognitive portion of computer-generated forces has been accomplished using several techniques including classical artificial intelligence (AI) techniques such as SOAR ("taking a State, applying an Operator, And generating a Result"), other AI formulations such as FuzzyCLIPS and Modular Knowledge Acquisition Tool (M-KAT), adaptation of analytical military models such as Suppressor, and specialized CGFs such as Modular Semi-Automated Force (ModSAF) and Interactive Tactical Environment Management System (ITEMS). This paper overviews these cognitive modeling techniques focusing on 14 specific features associated with intelligent agents.				
14. SUBJECT TERMS Artificial intelligence; AI; CGF; Computer-generated forces; Expert systems; Fuzzy logic; Intelligent agents;			15. NUMBER OF PAGES 31	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION ABSTRACT UL	

CONTENTS

	Page
INTRODUCTION.....	1
1. SOAR/IFOR.....	2
2. ModSAF Task Frames.....	7
3. AFIT Automated Wingman.....	8
4. Hughes Research Laboratory CFOR.....	10
5. Suppressor (Force Level Simulation).....	14
Software Requirements.....	16
Hardware Requirements.....	17
6. ITEMS.....	17
CONCLUSIONS.....	19
BIBLIOGRAPHY.....	20

Figures

Figure
No.

1	IFOR/SOAR System.....	5
2	AFIT Automated Wingman System.....	11
3	CFOR System.....	13

PREFACE

This work was accomplished by Hughes Training, Inc. (HTI), Training Operations, for Armstrong Laboratory, Human Resources Directorate, Aircrew Training Research Division (AL/HRA) at Williams Gateway Airport, Mesa, AZ.

The effort was conducted under Work Unit 2743-25-06, Flying Training Research Support, Contract No. F41624-95-C-5011. Laboratory contract monitor was Mr Daniel H. Mudd; laboratory technical monitor was Dr Herbert H. Bell.

INTELLIGENT AGENTS FOR COMPUTER-GENERATED FORCES

INTRODUCTION

This paper presents an overview of six separate techniques that have or are being applied as intelligent agents for computer-generated forces (CGF). These include:

1. SOAR ("taking a State, applying an Operator, And generating a Result)/IFOR (Intelligent Forces),
2. Modular Semi-Automated Force (ModSAF) Task Frames,
3. Air Force Institute of Technology (AFIT) Automated Wingman,
4. Hughes Research Laboratory Command Forces (CFOR),
5. Suppressor/Force Level Simulation (FLS), and,
6. Interactive Tactical Environment Management System (ITEMS).

The goal of these applications is the development of human-like agents for populating interactive distributed environments. It is planned that these agents will require minimal or no human operator intervention unlike current semi-automated forces.

Evaluation of each technique includes the following points and issues:

- a) Brief introductory description of technique and program application
- b) Description of the continuous operation of the entity and its weapon systems in a dynamic environment
- c) Description of real-time operation
- d) Degree of autonomy/amount of human intervention required of the technique
- e) Type of goal-directed behavior
- f) Learning capability
- g) Operation in conjunction with a higher level agent
- h) Coordinated behavior as a CGF member
- i) Selectable competency level capability
- j) Number of CGF entities that can be handled simultaneously
- k) Ease or difficulty in using the artificial intelligence (AI) technique
- l) Interface to ModSAF
- m) Additional comments where applicable
- n) Points of contact

1. SOAR/IFOR

a) SOAR (taking a State, applying an Operator, And generating a Result) is an expert system-based architecture that provides the replication of human behavior. It accomplishes this by recognizing that the problem space for this application is simply a set of states and operators that may be executed in the attempt to reach a goal. It has been continually developed over 12 years with 100 different users and 50 applications in several disciplines and represents a classical rule-based expert system. It has the ability to: (1) perform on a wide range of tasks including routine and difficult open-ended problems; (2) utilize many different problem-solving techniques required for these tasks; and (3) learn about the tasks and its performance on them. It is the basis of Newell's Unified Theory of cognition (Newell, A., 1990, Unified Theories of Cognition, Cambridge, MA, Harvard University Press).

Of particular interest to the simulation and training community is the Advance Research Projects Agency (ARPA)-funded program Intelligent Forces (IFOR/SOAR) with a consortium of university researchers including the University of Michigan Artificial Intelligence Laboratory, University of Southern California Information Sciences Institute, and the Carnegie Mellon University School of Computer Science. The What If Simulation System for Advanced Research and Development (WISSARD) program maintains a testbed of the IFOR software for purposes of testing and demonstration at Oceana Naval Air Station in Virginia Beach, VA. Their work over the last four years has been to develop SOAR-based agents that correspond to a computer-generated entity such as a pilot of a fighter plane or an electronics warfare officer. This includes a complete domain analysis for fixed-wing aircraft such as analysis of simulation exercises, actual exercises, and use of subject matter experts. The overall goal is automated behavior that is indistinguishable from human behavior. They have initially concentrated on fixed-wing aircraft but have recently expanded to rotary-wing agents. ModSAF, which interfaces to SOAR, is being used for vehicle dynamics, and the majority of sensors, weapons, and communications systems. IFOR/SOAR provides the pilot's perception of the synthetic battlespace (sensory information that would be obtained from the cockpit environment, e.g., radar displays, messages, vehicle status indicators and visual sightings out the cockpit) with a virtual cockpit and the cognitive knowledge of the agent. The SOAR agents send out commands to ModSAF that control the vehicle's motion, radar, weapons, and radio. ModSAF also provides the interface to the Distributed Interactive Simulation (DIS) network. They have developed a working interface between SOAR and ModSAF that was demonstrated in the Synthetic Theater of War-Europe (STOW-E) exercise. Currently, updates and enhancements are being done for involvement in STOW 97. Several papers have been published on the existing system and future enhancements. These papers are listed in the Bibliography.

The following analysis is based on the results and current work on IFOR/ SOAR.

b) IFOR/SOAR uses the concept of a "virtual cockpit" meaning that the agents have an interface that supports the type of interaction of a pilot in the cockpit of the aircraft. This provides perception of the world via state values of the vehicle such as sensed by instruments, and visual and sensor imagery, and isolates the IFOR/SOAR agent from the underlying simulation environment such as vehicle dynamics, network protocol, etc. The various agents can be grouped as sections (two aircraft) or divisions (four aircraft). Interfaces for IFOR/SOAR have specifically extended the standard suite of ModSAF sensors and weapons, such as Continuously Computed Interception Point (CCIP), waypoint computer, and precision guided weapons. Agents can fly the following missions in complex synthetic environments:

- 1) BARCAP
- 2) Close Air Support
- 3) Strategic Attack
- 4) MiGSweep

In addition, there are agents that act as air controllers during the missions. Current work should expand the automated synthetic pilots for the majority of air-to-air, air-to-ground, and support missions.

c) Real-time operation is accomplished by calling each SOAR agent (F-18 pilot, F-16 pilot, etc.) at each ModSAF simulation cycle. Multiple independent agents can be run on a single UNIX work station or on many different machines on a network. It is important to note that a single agent cannot be distributed across multiple machines. Agents can share data via simulated radio transmissions.

A major upgrade to enhance real-time application is version 6 of SOAR that is written in C language rather than version 5 which was written in LISP.

d) Once the agents have been assigned their missions, they will fly the missions without human intervention. Enhancements will include dynamic human intervention such as selecting new waypoints.

e) One of the key features of SOAR is its goal-directed behavior. Tasks and goals are represented as operators working on the problem space. This is done by employing a succession of operators that are applied to the existing state and guide the resulting transformation by rationality: If an agent has knowledge that an operator's application will lead to one of its goals, then the agent will select that operator to get to that goal. SOAR's rules are similar then to an associative memory, where information of the actions of the rules are recalled whenever the goal conditions of the rules match. This information resides in the long-term memory and is comprised of rules relating to doctrine, standard procedures, and lessons learned.

f) SOAR currently has one primary form of learning known as chunking. It is important to note that the term "chunking" is not used in the classical cognitive psychology sense of grouping data to learn. Learning results from detecting a lack of knowledge which occurs when an impasse or indecision is reached between two operators. When this occurs, the SOAR architecture automatically develops a subgoal of eliciting further knowledge needed to continue processing, i.e., searching out other options not necessarily associated with the primary goal.

SOAR/IFOR currently does not use learning. The reason is that the system is considered to be optimally expert and can learn no more. A more practical reason is the fact that debugging of the SOAR programs is difficult with learning. To incorporate learning in the SOAR/IFOR would require significant changes to the existing system. A key area of SOAR research is developing new learning strategies such as interpreting another person's gestures, robust general learning theories, means-end analysis, and effects of team behavior.

g) A group of SOAR agents, such as a section or division, will coordinate its overall behavior with the available controllers (Air Intercept Controller, Ground Intercept Controller, Forward Air Controller, Direct Support Center, Fire Support Center, Tactical Air Command Center, and Tactical Air Direction). These controllers can relate flight information, permission to continue the mission, or changes to the missions. The ability to communicate with other command forces is currently limited to these areas but will be expanded in the future. One of the additions will be the use of the Command and Control Simulation Interface Language (CCSIL) to interact with all SOAR/IFOR agents. This system has been

developed within the CFOR program which will be discussed later in this paper. Basically, this system allows a commander to prepare a five-paragraph order on a real-world command/control system with control intelligent agents in the simulation. The system takes the order and transforms it into simulation-relevant commands.

h) The IFOR/SOAR system provides coordinated behavior with other agents. The architecture provides several cognitive features which are related to military missions such as following a flight plan, planning attacks, employing weapons, and managing fuel. More general coordinated features include communication with other agents, modeling the behavior of other agents, being able to explain another agent's actions and general problem-solving techniques in a coordinated manner.

IFOR/SOAR agents coordinate their activities through a combination of overall long-term knowledge (results of training, standard operating procedures, doctrine, and tactics) and explicit communications, both verbal and nonverbal. Just as human pilots must use a medium outside their normal input/output (I/O) channels, SOAR agents communicate between each other using radio or digital transmissions. ModSAF has a generic radio interface using DIS radio Protocol Data Units (PDUs). Messages are generated by examining the active lists of rules which are turned into character strings by the SOAR ModSAF Interface (SMI). Verbal communications between agents are radio messages with content that is similar to pilot jargon. IFOR /SOAR currently has almost 100 templates of these messages.

Agents can fly in a number of formations and can dynamically break into smaller units. Most of the engagements are beyond visual range. Other coordinations are the ability to overlap radar coverage in the battlespace and coordination of weapon employment during air-to-air and air-to-ground engagements.

i) Currently, the system does not implement various competency levels although there are methods that could be applied. To date, the emphasis of IFOR/SOAR has been on gaining competency for a wide variety of missions. The main way that variable competency would be achieved is by having different rules sets loaded for different competency levels. Further factors being considered that give individual differences in pilots are fatigue, pilot workload, and attention span.

j) From six to ten agents can be processed simultaneously, stand-alone, on a 440 SGI Indy Unix machine. From experience in STOW-E, this number with network overhead dropped to four agents. The IFOR/SOAR program plans to use feature management techniques which will allow IFOR/SOAR to concentrate on subsets of the situation that are most relevant to the current tactical situation, thus decreasing processing.

k) SOAR provides a fixed architecture to provide rule matching, decision making, procedural analysis, support, and chunking. The long-term knowledge is made up of rules developed by the domain analysis such as tactics, plans, properties of aircraft, weapons, sensors, etc. Expansion of the domain to other military missions other than those included in IFOR/SOAR would require adding new associations and rules. This is done using existing SOAR Application Program Interfaces (API) to edit and manage new rules and other knowledge.

A noted deficiency is the mission setup. Although planned for the future, graphical interface tools are not available to set up the missions. Required editing of intermediate data structures is difficult, time-consuming, and prone to error. Ideally, the goal will be to give the user the touch and feel of the actual mission brief documents that define an agent's specific mission. One format being considered is the

Navy 9/12 debrief used to specify missions. Modification of ModSAF parameters for vehicle, weapon, and sensor parameters requires editing of data tables off-line using the existing ModSAF editor.

1) IFOR/SOAR has an established interface to ModSAF. Figure 1 shows the overall IFOR/SOAR architecture for one UNIX processor on the DIS network. Software consists of three components integrated within the same UNIX process: SOAR agents, SOAR-ModSAF Interface (SMI), and ModSAF. ModSAF provides the DIS interface. All software is in C and the SOAR agents are called at each ModSAF cycle providing system synchronization. The use of ModSAF as the master synchronization seems reasonable since ModSAF controls the synthetic battlespace while SOAR agents are agents of that world. Communication between modules is done using C function calls thus decreasing communication overhead. The SMI makes all the calls to the underlying ModSAF functions to gain access to the majority of vehicle, sensor, and weapon states. Separation of the vehicle simulations is not clear-cut, thus the SMI fills in any gaps to provide the full cockpit simulation to the SOAR agents. The SMI does not use ModSAF tasks or task frames but relies on lower level commands and functions which give the agents direct control of their own behaviors more closely corresponding to cockpit controls.

Acquisition of knowledge:

- Interview human experts
- Analyze human behavior from simulators
- Analyze human behavior from actual flights
- Analyze human reasoning

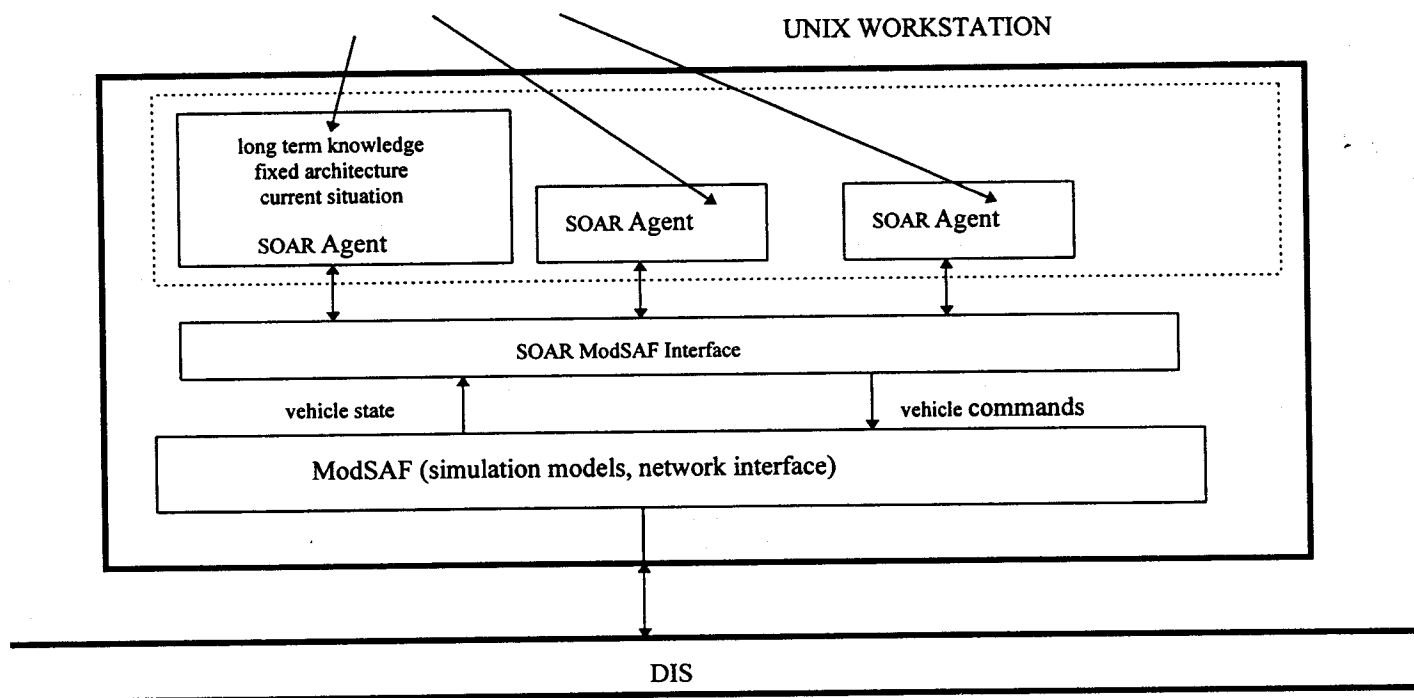


Figure 1
IFOR/SOAR System

Long-term knowledge or rules contain tactics, doctrine, plans, standard operating procedures, properties of aircraft, sensors, weapons, etc. This is derived from the acquisition of knowledge. The fixed architecture contains fundamental portions of SOAR's unified cognitive model including rule matching, decision making, and chunking. The current situation involves evaluation of necessary goals, proposed operators, and current plans as well as models of other agents.

Vehicle commands to ModSAF are those commanded by the SOAR agent in the attempt to meet his goals. These could include such parameters as desired altitude, desired headings, desired radar mode, etc. Vehicle state feedback to the agent is the actual state of the vehicle that the pilot perceives through his sensory modalities.

m) This program is one of the first to integrate a classic established AI package with a real-time simulation and use it in an operational exercise. It has shown limits in the number of agents that can be supported as well as difficulty in setting up missions and creating complex agents using file edits. Experience with SOAR has shown that it is difficult to set up even simple cases of cognitive modeling. Its performance, according to demonstrations at the Naval Air Station Oceana involving simulated air-combat engagement against expert human pilots, was well received although the maneuvers were limited. Upcoming enhancements will help to rectify these issues.

In order to build the most general systems and complex behaviors, other techniques such as means-end analysis, planning, extensive learning, and use of natural language is necessary.

Several enhancements have been presented such as feature management of the scenario, better graphical user interfaces, and more realistic mission statement inputs. Another enhancement is the use of natural language processing where SOAR agents directly communicate to humans. Agents will need to understand and generate natural language integrating it with other agents' tasks. Finally, as complex SOAR agents are created, it will be necessary for each to explain its overall behavior. A debrief system is under construction which will allow an agent to be debriefed after a mission.

n) Points of contact for IFOR/SOAR and WISSARD are:

ARPA IFOR/WISSARD
Len Breslow
Naval Research Laboratory, Code 5515
Washington, DC 20375-5337
(202) 404-7736
breslow@aic.nrl.navy.mil

SOAR/ IFOR
John Laird
Artificial Intelligence Laboratory
University of Michigan
1101 Beal Ave.
Ann Arbor, MI 48109-2110
laird@umich.edu

2. ModSAF Task Frames

- a) ModSAF provides the capability to create and control entities in a simulated battlefield. The goal of ModSAF is to replicate the behavior of these entities adequately for training and combat development purposes.
- b) ModSAF allows for continuous operation of its entities and their associated weapon systems in a dynamic environment.
- c) ModSAF operates using a variable time tick that periodically updates each simulated entity. As the simulation load increases, the update frequency decreases and performance can gradually degrade. Numerical integrations and other processes using the delta time are done with the variable time length defined by the required frequency. There is no known mechanism that ties ModSAF time to "wall clock time." Real time is achieved by assuring that the load on the simulation computer is not excessive, i.e., if there are too many vehicles on the computer or network traffic is too great, it is necessary to move some of the players to another computer to maintain real-time operation.
- d) The user must create vehicle missions via task frames assigned in the Unit Operations Editor. Each task frame has an associated set of parameters that must be specified. Examples of task frames include such tasks as attack, withdraw, and hasty occupation of battle position. Once the vehicle mission is defined, the user can let it run autonomously or can intervene as needed throughout the mission.
- e) ModSAF vehicle behavior is goal-directed, but not in the traditional AI sense. For example, the user can assign a ground attack target task frame to a vehicle, and the vehicle will then take the necessary steps to approach and attack the target. However, much of the information needed for the vehicle to accomplish the goal is specified on a graphical user interface (GUI) editor. The ModSAF task code then uses this information to carry out the vehicle mission.
- f) ModSAF CGF entities do not have a learning capability.
- g) ModSAF vehicles can operate in conjunction with a higher level agent. ModSAF 1.5.1 currently contains flights of two, three, and four F-16s in which one of the aircraft is designated as the leader and the others as followers. The leader and followers exhibit different behaviors based on the situation. If a leader is killed, another vehicle in the group is designated as the new leader and inherits the leader's behavior.
- h) Behavior is coordinated between team members. In the defined flights of F-16s, one vehicle is designated as the leader and the others as followers. For some tasks, the leader has the more complicated behavior and the others simply follow at a defined distance. In other tasks, such as a split to attack a target, a second vehicle will be designated to lead the second half of the group and will then have the same behavior as the first leader. In general, if a leader is killed, another vehicle in the group is designated as the new leader and inherits the leader's behavior.
- i) When a vehicle is created in ModSAF, the user can assign a competency level from 0 to 1 (0 novice, 0.5 average, 1 expert). This competency factor does not affect representative human processes (perception, motor behavior, cognition, communication) since ModSAF does not directly model these, but it does affect weapon delivery accuracy and damage assessment.

j) The number of CGF entities that can be handled simultaneously by ModSAF depends on the mission, entity types, terrain, density of entities on the terrain, and computer and operating system configurations. A benchmark test done on ModSAF 1.0, running in the pocket configuration (simulation and GUI both running on the same machine) with no network connection, showed that the system could handle 36 tanks that were constantly moving and firing on targets. The machine used was a Silicon Graphics Indigo with a 50 MHz R4000 processor. It was estimated that the computational resources needed for one tank are approximately the same as the resources needed for 0.5 fixed-wing aircraft (FWA), meaning that one could expect to run 18 FWA in this stand-alone pocket configuration. In a networked environment, this number decreases dramatically to 4-5 FWA vehicles.

k) ModSAF Task Frames are not a true AI technique, they are simply behavioral logic that is implemented in code. This technique requires a programmer working closely with a subject matter expert (SME) to correctly implement the desired behavior. This method can be very complex and tedious for someone who is not familiar with the ModSAF architecture.

l) ModSAF can be run on a variety of platforms including: SGI Indigo, Indigo 2, or Indy (with 96 MB swap space), SPARC 10 (with 65 MB swap space), Mips 3000 (with 180 MB swap space), or IBM RS 530 (with 150 MB swap space). ModSAF can easily run in the pocket configuration on a 1.2 GB hard disk. The ModSAF software is written in C.

m) ModSAF is capable of implementing vehicle behavior using task frames, but this is not a true AI technique. Task frames are merely code written to implement behavior based on consultation with an SME. Depending on the complexity of the desired behavior, development of the task frames/tasks can be straightforward or very complicated.

n) Point of contact for ModSAF is:

Ellen Mallery
Hughes Training, Inc.
P. O. Box 1237
Binghamton, NY 13903-1237
(607) 721-4360

Questions about ModSAF can be addressed to Loral Advanced Distributed Simulation through the ModSAF reflector. Everyone in the ModSAF community is encouraged to participate in the discussions and answer questions. To register with the reflector, write to listproc@sc.ist.ucf.edu. The text of the message must be SUBSCRIBE MODSAF firstname lastname. (Your e-mail address will be pulled out of the e-mail header.) To unsubscribe from the reflector, write to listproc@sc.ist.ucf.edu with unsubscribe MODSAF in the body of the e-mail. Leave the Subject field blank. Also, be sure to send this request from your registered e-mail address. In addition, reflector traffic is maintained on-line for 60 days. You can access these e-mail messages via World Wide Web. Open URL, <http://www.sc.ist.ucf.edu/~SC/confs/MODSAF>.

3. AFIT Automated Wingman

a) This program which is relatively new has the goal of incorporating an intelligent agent for a wing of a lead manned simulator. The program is being supported by the Electronic Systems Command and is the basis of a recently released Masters thesis from AFIT. The Automated Wingman is a semi-automated,

computer-generated aircraft which operates under the control of a lead simulator. Human behavior is provided with a fuzzy expert system and a voice interface. The fuzzy expert system is configured around a set of hierarchical goal-oriented databases. Fuzzy logic represents a form of approximate reasoning. It provides an excellent mechanism for the recognition of knowledge as well as the inherent ambiguity and uncertainty of that knowledge. These databases provide fuzzy set logic using standard linguistic variables that define control action for the wingman. Voice interaction allows the lead to direct activities of the wing. Currently, the wing flies in a tactically correct formation with the lead and does not fight.

The heart of the system is FuzzyCLIPS, a modified version of the expert system CLIPS developed by NASA. CLIPS is being used by over 5,000 users in various applications. FuzzyCLIPS is an enhanced version of CLIPS developed at the National Research Council of Canada to allow the implementation of fuzzy expert systems. Modifications made to CLIPS contain the capability of handling fuzzy concepts and reasoning. It enables domain experts to express rules using their own fuzzy terms and allows any mix of fuzzy and normal terms, numeric-comparison logic controls, and uncertainties in the rule and facts. Fuzzy sets and relations deal with fuzziness in approximate reasoning, while certainty factors for rules and facts manipulate the uncertainty. Use of the above modifications is optional, and existing CLIPS programs still execute correctly.

b) The automated wingman will use higher fidelity aerodynamic, weapon, and sensor models rather than ModSAF models. They have used the aero model designed by Cooke at the Naval Postgraduate School and are currently evaluating various other models for application. The Automated Wingman is intended to provide DIS-compatible simulators with one or more unmanned but intelligent wingmen.

c) The system runs real time at 30 Hz update rate on a Silicon Graphics Onyx Reality Engine. The language is C++.

d) It is intended that little human intervention other than voice commands from the lead in the manned simulator will be required.

e) The automated wingman requires knowledge of which maneuvers to perform to achieve certain goals. The system defines a goal and uses a planning system to guide its actions toward that goal. These goals are hierarchical in nature. Using this structure, the Automated Wingman can employ various tactical maneuvers to achieve the goal. Further research is required to define the total goal hierarchy for such items as weapon delivery based on subject matter experts and domain analysis.

f) Learning capability will not be included.

g) The Automated Wingman is able to receive and execute a number of voice commands from the lead simulator. At this time, there are 48 commands. These commands are basic maneuver commands, commands to employ tactics, target designation, flight mode, etc. and must be executed in a tactically and doctrinally correct manner.

h) Coordinated behavior will have the wingman performing tactically and doctrinally correct maneuvers with the lead aircraft.

i) Currently, selective competency level is not included but planning data and fuzzy rules could be modified to achieve this. This is to be added in the future.

j) The current application has a manned lead simulator and a computer-generated wingman.

k) A key part of the fuzzy expert system is the ability to modify and change the set of production rules. In order to accomplish this, FuzzyCLIPS supports a construct in which linguistic variables can be represented for production rules. The fuzzy membership grades also can be easily constructed using the FuzzyCLIPS editor. Since there has been considerable development of FuzzyCLIPS, this technique provides a very user-friendly system interface.

l) The proposed system will not interface to ModSAF at all. Access to DIS and vehicle, weapon, and sensor simulations will be derived from other AFT sources. They will use a DIS interface known as Object Manager which is Silicon Graphics platform dependent. The system is UNIX-based on a Silicon Graphics platform written in C++. The software architecture is highly object-oriented using Rumbaugh OOM techniques. Figure 2 shows the basic functions of the system. The AFIT Virtual Cockpit was a result of an ARPA program to develop a low-cost, DIS-compliant manned flight simulator.

m) This program is in initial development and will continue to mature in the near future. The technique as planned offers a different approach of using fuzzy systems and planning techniques. It is based on a strong, well-used, and documented core of FuzzyCLIPS. One limitation is the use of only a manned simulator lead and wing. This program should be revisited in the near future as it matures and adds more fighting capability since the concept shows promise.

n) Point of contact for the AFIT Automated Wingman is:

LtC Martin R. Stytz, Ph.D.
Associate Professor of Computer Science and Engineering
Department of Electrical and Computer Engineering
Air Force Institute of Technology
AFIT/ENG, Bldg. 640, Room 218
2950 P Street
Wright-Patterson AFB, OH 45433
MSTYTZ@AFIT.AF.MIL

4. Hughes Research Laboratory CFOR

a) Command Forces (CFOR) simulation incorporates the modeling of battlefield command and control (C2) into virtual simulation. It extends the DIS architecture with the addition of a new class of DIS command entities. C2 stations communicate with computer-generated entities through the Command and Control Simulation Interface Language (CCSIL) developed by MITRE Corporation. The overall CFOR project is a part of the Synthetic Theater of War (STOW) program which is an Advanced Concept Technical Demonstration that is sponsored by the Advanced Research Projects Agency (ARPA). CFOR is under development by a team of government contractors and laboratories including Hughes Research Laboratory (HRL). The CFOR architecture allows automated command entities and Semi-Automated Forces to work together realistically. HRL is currently developing a platoon and company commander for the Marines and last year developed an Army company commander. The HRL CFOR employs symbolic reasoning and concurrent control in the development of the Canonical Commander Model. The core inference tool for HRL CFOR is a Hughes product called Modular Knowledge Acquisition Tool (M-KAT). The goal of the Hughes program is to develop combined and individual combatant capability for the Marine Corps component of STOW-97.

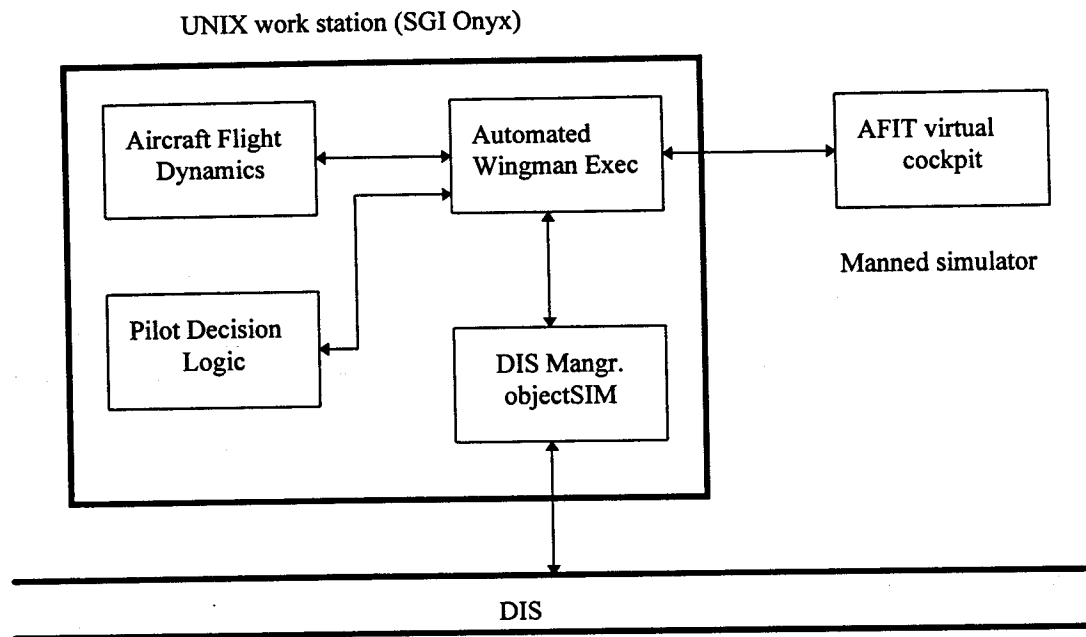


Figure 2
AFIT Automated Wingman System

b) The CFOR uses the ModSAF vehicle, sensor, and weapons models. Several of the databases were modified to reflect unique Marine Corps requirements. A key part to CFOR's operation is the ability to use the CCSIL providing interaction with a command entity. CCSIL was developed by MITRE to incorporate interoperability between command entities (decision makers) and platform entities in the DIS environment. This allows synthetic commanders at simulated command and control stations to interact via orders and directives to computer-generated entities and expect them to react realistically. This includes use of sensors and weapons for maneuver and fire in the synthetic battlespace. CCSIL includes a set of various military messages that can be used to command the virtual entities. The format is in the normal military five-paragraph order. The description of the continuous operation is as follows:

- 1) A Battalion order is passed via CCSIL to the Company level.
- 2) The company level CFOR parses the order into goal and constraint measures.
- 3) A course of action based on doctrine and tactics is defined.
- 4) Operational orders are sent to platoon level.
- 5) The task frames of ModSAF are used to accomplish the goal of the order.
- 6) Situation and fragmentary reports are sent through the command chain and output via CCSIL to the Battalion level for evaluation of the current state.

c) Real-time operation is accomplished by using ModSAF for the master clock for the CFOR execution. In addition, the CFOR system interfaces to the CFOR infrastructure (another MITRE product) allowing interfaces to simulated C2 equipment and use of CCSIL for receiving and transmitting messages in real time.

d) The degree of human intervention is based on the activity of the command entity through CCSIL. Once orders are given, the simulation runs using the task frames of ModSAF. The normal functions of ModSAF such as repositioning are available if required for any reason.

e) The Hughes CFOR uses symbolic reasoning modules that assess the blue-and-red situation as well as terrain analysis from the point of view of the commanded entities and initial conditions. Based on commands from superiors and requests from subordinates, the CFOR analyzes the mission to identify explicit and implicit tasks based on tactics and doctrine in order to meet mission objectives. These tasks are combined into a best course of action which are specific goals such as keep moving, minimize losses, etc. The symbolic rules are based on fuzzy variables which define the degree to which rules fire. The membership grades for the fuzzy variables are easily defined using M-KAT in a spreadsheet format. It is possible that there will be competing as well as coexisting goals. HRL has proposed the concept of concurrent control which arbitrates these goals based on a weighting system (set up initially off line) to define a final command for the synthetic forces. For example, the weighting may be low for "minimizes losses" for an aggressive commander. The execution is continuously monitored, coordinated, and updated.

f) There is no mechanism for learning in a formalized manner. Situation reports and fragmentary reports allow the synthetic commander to react to the situation based on his knowledge and perception of the battlespace.

g, h) Currently the synthetic commander is at the battalion level. The commander can respond and react to the following orders:

- Operation orders
- Fragmentary order
- Execute directive
- Report-Request

Using these messages, the commander can command and control his company through a military scenario in the virtual battlespace. The CFOR Armor Company is composed of three tank platoons plus the commander's and executive officer's tank. The CFOR (using the Canonical Commander Model) determines the company course of action and defines task frames for ModSAF to perform fundamental fire and movement tasks. This system provides a complete chain of command from battalion down to platoon level using simulated C2 equipment. The Hughes CFOR system is capable of higher levels of command control echelons but is limited by ModSAF aggregation capabilities.

i) Since weighting of the goals can be easily accomplished, the ability to change levels of competency can be done as well.

j) Currently the number of entities supported is limited by ModSAF. The number that has been used for recent demonstrations is from 25 to 30 entities. This corresponds to three tank platoons.

k) The use of the M-KAT tool to define databases, fuzzy functions, and inference rules provides an easy-to-use system for the operator. It is based on spreadsheet format and can be customized to different formats that an SME may choose.

states) reasoning techniques in conjunction with behavior-based (goals) control techniques provided a large variety of maneuver tactics for computer-generated F-14 entities. A key component of this approach is the development of the cases which consist of state variables that describe the motion and geometry at a particular instant in time. These cases are defined off-line using subject matter experts analyzing different possible geometries and conditions. From matching these cases with the real-time simulation, various behaviors will result (maintain formation, avoid threat, etc.). Since there may be more than one specific goal that is invoked to some degree, arbitration logic based on weighting criteria must be employed for commanded aircraft inputs.

n) Point of contact for the Hughes Laboratory CFOR and IFOR is:

David Y. Tseng
Hughes Research Laboratory
3011 Malibu Canyon Road
Malibu, CA 90265
(310) 317-5677
tseng@aic.hrl.hac.com

5. Suppressor (Force Level Simulation)

a) SUPPRESSOR is a DoD general purpose event-stepped, mission-level simulation model controlled and distributed by USAF ASC/XREM, Wright Patterson AFB (ASC). It has traditionally been used by the analytical community for survivability analysis and Cost of Operation and Effective Analysis (COEA). With over 60 user beta sites, SUPPRESSOR is well accepted by this community. HTI has used SUPPRESSOR (Force Level Simulation) for training by modifying the model to run in real time and be interactive with outside entities via the DIS Protocols. One reason the use of SUPPRESSOR for training has been successful is because it provides the user the flexibility to model realistic player behavior (1). The existing interface to SUPPRESSOR could be modified to capitalize on this quality and use it to provide a realistic behavioral agent for ModSAF. The following text in this section will describe the capabilities currently in SUPPRESSOR. Any additional capabilities that may be required would be added in compliance with the architecture established by the SUPPRESSOR decision-making algorithms.

b) The usage of SUPPRESSOR for training has been as an autonomous system with minimal instructor control. SUPPRESSOR provides for database input of tactics for lethal assignment, lethal engagement, nonlethal engagement, coordination, emission control, launch, and movement tactics. Lethal assignment tactics define the criteria (rules) for a commander to assign and de-assign a perceived target to a subordinate. These tactical definitions also provide for decentralizing control to a subordinate. Lethal engagement tactics define the criteria for placing targets on a queue for possible engagement. Additionally, they define the tactics for actual initiation and termination of lethal engagements. Emission Control tactics define the criteria for turning on/off systems. Launch tactics define the conditions under which a commander will issue launch orders to subordinates. Movement tactics refer to rules employed to reactively maneuver; decisions of whether to use terrain following, terrain avoidance, and threat avoidance; contingency plans for reactively maneuvering player locations; employment of plan patterns; employment of plan profiles; and employment of plan aspect tables which define 3-D movement relative to a target based on aspect. Non-lethal engagements are tactics related to sensor and communication jamming. Coordination tactics define the criteria for the use of 3-D zones,

decentralizing to subordinate control in case of loss of communication, time frame for intelligence reports within a command chain, and the criteria for sending message/sensor reports within a command chain.

c) Each SUPPRESSOR event has a time tag associated with it. The processing of events causes the scheduling of subsequent events. Real time is maintained by synchronizing the execution of the events with wall clock time. The real-time control allows the system to be frozen as well as initialized to a new scenario (1).

d) No human guidance or intervention is required for player interaction with other entities (both humans and other CGFs) in the environment. All tactics are defined during scenario/player development by constructing an ASCII file of formatted instructions via a text editor or the X Windows User Friendly Interface (UFI).

e) SUPPRESSOR demonstrates goal-directed behavior through the implementation of tactics and orders defined in an ASCII database. This goal-directed behavior is carried out by SUPPRESSOR's most important system--a player's thinker. A SUPPRESSOR player will notice, digest, and react by servicing a mental processing queue. An item gets added to the processing queue from a physical event (communication, movement, sensing, shooting, or status change), a "wakeup to think about" event scheduled by the thinker, or the continuation of some activity by the thinker. While a thinker system is processing an entry or parallel entity, it is considered busy. This, coupled with the fact that each type of item in the queue has a TIME_TO_THINK data item which prevents the processing of the entry for a user-defined period of time, creates a human characteristic in that the SUPPRESSOR thinker system can become overloaded (4).

The decisions that a thinker makes are dependent on the player's perception of his environment. Perceptions are a player's view of reality based on what is determined from his own sensor inputs as well as what is communicated to that player from other players. Perceptions are maintained in the player's memory for a user-defined amount of time. The concept of perceptions allows the model to separate ground truth from player knowledge. Players will know only what they have a right to know. Additionally, players will make decisions based on the perceived environment, rather than the actual environment--whether the perception is correct or not. This creates a situation of realism which can be explained with an example. If a ground player with a radar is tracking an air target, the ground player will have both an indication on his sensor as well as a perception of the air player. If the air target flies behind a hill, the sensor indication will go away. However, the perception will continue and follow a path that represents the direction and speed at which the air target went behind the hill. Now if the air target stops, slows down, or speeds up, the ground player's perception of where the air target is and where the air target actually is will differ.

f) SUPPRESSOR has no method of learning but provides to the user text and binary outputs for analyzing what happened. The particulars of each event are captured. HTI has created a Debriefier that allows the user to graphically play back what is captured in the binary files.

g) The types of information communicated between commanders and subordinates via their communication systems are as follows: Weapon Assignment, Assignment Status, Cancel Weapon Assignment, Mode of Control Change, Move Order, Intelligence, and Queuing Message. A Weapon Assignment is a message from a commander to a subordinate that assigns a perceived target to the subordinate. There are five Assignment Status type messages. A subordinate will indicate that a weapon reload is finished. A launched player will communicate that it has started moving. A subordinate will

indicate to a commander that it is nonoperational. A player will tell his commander when he has started an engagement. Additionally, a subordinate will communicate to the commander about his perception of an assigned target. The Cancel Weapon Assignment type of message is sent from a commander to cancel a target assignment. The Mode of Control Change message is sent to a subordinate to tell him to go into weapons free or tight. The Move Order is used by a commander to launch a subordinate. The recipient of an Intelligence Message can be a commander, subordinate, or peer, and the contents includes information about perceived players. Finally, a queuing message is sent from the commander to tell the subordinate to change its heading (2).

h) Coordination of player actions relative to one another can be modeled within SUPPRESSOR. Coordination is a combination of communication between players and the individual players' tactics. Both sensor reports and message reports can be communicated between peers. These reports are sent by players to inform others within a command chain about targets they can see with their sensors or are told about via messages. The individual players' tactics must utilize this information to determine the most appropriate action to take.

i) Competency level is currently designed by the database developer. To select between competency levels during real time, one would change the various values within SUPPRESSOR for thinking rates, firing rates, and sensing rates. Specific examples of values which would affect competency within SUPPRESSOR are as follows: time to think, sensing mode rate, time intervals for intelligence reports, time to transmit communications (by type of communication), maximum number of simultaneous perceptions, time to react to a frequency change, firing time delays, and reaction times (2).

j) FLS/SUPPRESSOR's most recent usage on a program involved 32 players on a 4 CPU 40 MHz SGI 4D-340 with a DIS network and 3-D SGI Graphics Package running. The numbers should increase significantly on a newer machine with no graphics or network software being run by FLS/SUPPRESSOR. It is estimated that the number of players which can be supported by the proposed design will increase to well over 100.

k) The development of the player tactics is done via a UFI or by editing a text file. In one sense, the development of tactics may be easier than on some other systems in that actual source code does not have to be written. The UFI provides the user with on-line documentation as well as syntax checking (1). It prompts the user for required and optional data via an X/Motif graphical interface. However, to create a rule set that demonstrates realistic behavior requires someone knowledgeable in using the system as well as real-world player tactics. A training course from an experienced database developer who has used SUPPRESSOR for the training environment is strongly recommended.

l) ModSAF Interface

Software Requirements. The recommended interface for SUPPRESSOR to provide the intelligence for ModSAF players is to have two separate processes which communicate with each other via a shared memory interface. The information communicated would be at the level that permits each system to perform the functions for which they are best designed. For example, SUPPRESSOR has tactics to put a player into weapons tight or free; these are also selections for the operator on the ModSAF GUI. Therefore, the interface from SUPPRESSOR to ModSAF could be to put the player in tight or free. On the other hand, SUPPRESSOR could be used to go down to the level of telling ModSAF players which weapon to shoot at whom and when. The level of integration of the interface will directly affect the level of correlation required by the two systems.

SUPPRESSOR has tactics to tell when to turn on/off systems, when to change frequencies if communications are jammed, and where to point system antennas. It must be assumed that ModSAF will be expanded where needed to provide the lower level system characteristics sufficient to demonstrate realistic behavior. SUPPRESSOR could provide sensor, communication, and jamming simulations, but this design uses SUPPRESSOR for more than the behavioral agent.

ModSAF will have commander players for which SUPPRESSOR can provide the thinking process. The commands which will be communicated were covered above.

SUPPRESSOR control of ModSAF player movement can be done at two levels. SUPPRESSOR has reactive movement capabilities which take the form of move plans. Which move plan is executed is determined by the player's tactical logic. On one level, ModSAF could be sent the move plan SUPPRESSOR wants executed. On a lower level, the movement paths generated by SUPPRESSOR as the result of executing a move plan can become target way points for ModSAF players.

SUPPRESSOR is written in FORTRAN and the FLS interface software is in C and FORTRAN.

Hardware Requirements. Individually, both ModSAF and FLS/SUPPRESSOR can run on a one-CPU machine. There is no architectural problem with running them both on the same one-CPU machine. However, a more conservative approach would be to have a multi-CPU machine on which each would get a CPU. In either case, benchmarking is recommended. The current numbers available for each system are from runs on older machines.

Whether a single or multi-processor machine is chosen, it is recommended that the largest possible amount of memory is allotted (128 MB). Since the recommended configuration keeps each system as a separate process, each will have its own large memory requirement.

n) The point of contact for SUPPRESSOR and Force Level Simulation is:

Dennis Miller
Hughes Training Inc.
P. O. Box 1237
Binghamton, NY 13902-1237
(607) 721-4676

6. ITEMS

a) The Interactive Tactical Environment Management System (ITEMS) is a real-time tactical environment that can simulate the actions and behaviors of up to 100 players. ITEMS allows player behavior to be controlled using an inference engine. The user creates rules and rule sets using IF-THEN-ELSE logic which are interpreted by the inference engine and translated into parameter values which then invoke player behaviors within the ITEMS code. The rule sets can be very simple or very complex depending on the desired player behavior. A rule set can be a progression of straightforward IF-THEN logic or can invoke other defined rule sets.

b) The inference engine coupled with ITEMS provides continuous operation of 100 players and their associated weapon systems in a dynamic environment.

c) Although the inference engine runs asynchronously, it expects to operate with a simulation running in real time. This way the rules can be "fired" based on a certain time in the scenario or to allow coordination between players' actions.

d) Since the rules/rule sets are defined in advance and executed as needed during the scenario, there is no need for human intervention when the simulation is running.

e) The player behavior is goal-directed based on the implementation of the rule sets. The player will behave according to what is defined in the rule set conditions and consequences.

f) The inference engine does not have a learning capability.

g) The ITEMS inference engine allows for operation with two levels of higher level agents. Rules can be created for individual players, company commanders, and battalion commanders. There is no limitation on how many levels that could be implemented using the inference engine and rules. The only limitation in ITEMS is that there are no parameters yet defined for higher levels, i.e., there is no code written in ITEMS that could be invoked for levels of command above battalion level.

h) By using different rules for commanders, coordinated behavior among team members can be achieved. For example, a commander of a group of tanks could issue a retreat command to his subordinates. This merely sets a parameter which invokes the retreat action for the subordinates.

i) The inference engine can model competency levels only if the user defines the rule sets appropriately. A novice could be given a simple rule set to implement simple behavior while an expert player could be given a complex rule set that would take more complex situations into account.

j) The inference engine with ITEMS can handle 100 players simultaneously where the computer hardware is the limiting factor. The inference engine on its own could handle considerably more players. If the inference engine were to be integrated with ModSAF, the ModSAF processing (player dynamics, network traffic, etc.) and the computer platform would be the limiting factors.

k) The ITEMS inference engine has a user-friendly GUI that allows a user to combine conditions and consequences, using logical operators, into rules and rule sets. Since the rules are tied to simulation parameters, a programmer working with a subject matter expert is needed to initially define an adequate set of rule parameters. The user must keep in mind that any rules created may not produce the expected results initially because the simulation code ultimately dictates the players' actions. Thus, the user must be careful not to define rules that conflict with the code and must be certain to define rules adequately so that the desired player actions are invoked.

l) The ITEMS inference engine software consists of approximately 300 KB of FORTRAN code that runs asynchronously during runtime. The GUI and rules parameter processing code, consisting of approximately 550 KB of C code, is run off-line and is utilized during scenario definition. It is necessary to have the simulation parameters defined in global memory. The function of the rules parameter processor is to associate an English text rule with a corresponding global parameter in the ITEMS run-time code.

m) An inference engine is a relatively simple yet very powerful AI technique for controlling player behaviors in a simulation. Any parameter that is defined in the simulation code can be accessed and turned into an English text rule using a rules parameter processor. This allows the user essentially

unlimited ability to control player behaviors. However, the user should be cautioned to keep the rules relatively simple to avoid contradictions with the simulation code and to avoid writing "new code" outside of the simulation code via the rules/rule sets definition process.

n) Point of contact for ITEMS is:

Charles Smith, M/S 405
Hughes Training Inc.
P. O. Box 6171
Arlington, TX 76005-6171
(817) 695-8521

CONCLUSIONS

A number of different techniques have been considered for intelligent agents including classical AI, fuzzy expert systems, concurrent control, and planning. Each has advantages and disadvantages. Perhaps the best overall approach to ensure salability and extendability is a combination of all the techniques. The specific requirements of the application must be clearly understood to determine a specific approach. Considerations include command level, fidelity levels, number of players, real-world communications requirements, and computational resources. It appears that the most robust systems will include, in part if not all, fuzzy techniques which optimize the solution space mapping.

A key portion of the creation of intelligent agents is the domain analysis to define the necessary databases of knowledge and rules. This can be done by several methods, such as interviews with subject matter experts, and collection of data from manned simulations or actual exercises. Use of neural networks to determine clustering of data from exercises is a powerful tool to develop fuzzy rules. The domain analysis is a common task for all techniques which are usually labor intensive, i.e., costly. It seems logical that domain analysis could be a common task for all these techniques rather than each application performing the task.

BIBLIOGRAPHY

IFOR/ SOAR

1) Collected papers of the Soar/IFOR project (Spring 1994). Johnson, W. Lewis, Jones, Randolph M., Koss, Frank V., Laird, John E., Lehman, Jill F., Nielsen, Paul E., Rosenbloom, Paul S., Rubinoff, Robert, Schwamb, Karl B., Tambe, Milind, van Lent, Michael, & Wray, Robert.

This technical paper contains most of the papers published by this group between May 1993 and May 1994. It is available from the University of Michigan (CSE-TR-207-94), the University of Southern California Information Sciences Institute (ISI/SR-94-379), and Carnegie Mellon University (CMU-CS-94-134).

2) Collected papers of the Soar/IFOR project (Spring 1995). Johnson, W. Lewis, Jones, Randolph M., Koss, Frank V., Laird, John E., Lehman, Jill F., Nielsen, Paul E., Rosenbloom, Paul S., Rubinoff, Robert, Schwamb, Karl B., Tambe, Milind, Van Dyke, Julie, van Lent, Michael, & Wray, Robert.

This technical paper contains most of the papers published by this group between May 1994 and May 1995. It is available from the University of Michigan (CSE-TR-242-95), the University of Southern California Information Sciences Institute (ISI/SR-95-406), and Carnegie Mellon University (CMU-CS-95-165).

3) Johnson, W. L. (1994). *Agents that Explain Their Own Actions*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

4) Johnson, W. L. (1994). *Agents that learn to explain themselves*. Proceedings of the Twelfth National Conference on Artificial Intelligence. Seattle, WA.

5) Johnson, W. L., & Tambe, M. (1995). *Using Machine Learning to Extend Autonomous Agent Capabilities*. The Proceedings of the 1995 Summer Computer Simulation Conference, Society of Computer Simulation.

6) Jones, R. M., & Laird, J. E. (1994). *Multiple Information Sources and Multiple Participants: Managing Situational Awareness in an Autonomous Agent*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

7) Jones, R. M., Laird, J. E., Tambe, M., & Rosenbloom, P. S. (1994). *Generating Behavior in Response to Interacting Goals*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

8) Jones, R. M., Wray, R. E., van Lent, M., & Laird, J. E. (1994). *Planning in the tactical air domain*. In Planning and Learning: On to real applications, papers from the 1994 AAAI Fall Symposium (Technical Report No. FS-94-01). Menlo Park, CA: AAAI Press.

9) Koss, F. V., & Lehman, J. F. (1994). *Knowledge Acquisition and Knowledge Use in a Distributed IFOR Project*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

- 10) Laird, J. E., Jones, R. M., & Nielsen, P. E. (1994). *Coordinated Behavior of Computer-Generated Forces in TacAir-Soar*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.
- 11) Laird, J. E., Jones, R. M., & Nielsen, P. E. (1995). *Multiagent Coordination in Distributed Interactive Battlefield Simulations*. Abstract published in Proceedings of the International Conference on Multi-Agent Systems (ICMAS).
- 12) Lehman, J. F., Van Dyke, J., & Rubinoff, R. (1995). *Natural Language Processing for IFORs: Comprehension and Generation in the Air Combat Domain*. Proceedings of the Fifth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL, pp. 115-123.
- 13) Nielsen, P. (1995). *Intelligent Computer-Generated Forces for Command and Control*. Proceedings of the Fifth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL, pp. 211-218.
- 14) Rubinoff, R., & Lehman, J. F. (1994). *Natural Language Processing in an IFOR Pilot*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.
- 15) Schwamb, K. B., Koss, F. V., & Keirse, D. (1994). *Working with ModSAF: Interfaces for Programs and Users*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.
- 16) Tambe, M., Jones, R. M., Laird, J. E., Rosenbloom, P. S., & Schwamb, K. (1994). *Building Believable Agents for Simulation Environments: Extended*. Proceedings of the AAAI Spring Symposium on Believable Agents.
- 17) Tambe, M., & Rosenbloom, P. S. (1994). *Event tracking in complex multi-agent environments*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.
- 18) Tambe, M., & Rosenbloom, P. S. (1994). *Event tracking for An Automated Intelligent Agent*. Proceedings of Time 94: An international workshop on temporal representation and reasoning.
- 19) Tambe, M. (1995). *Recursive agent and agent-group tracking in a real-time dynamic environment*. In Proceedings of the International Conference on Multi-Agent Systems (ICMAS).
- 20) Tambe, M., & Rosenbloom, P. S. (1995). *RESC: An approach for real-time, dynamic agent tracking*. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI).
- 21) Tambe M., Schwamb, K., & Rosenbloom, P. S. (1995). *Building intelligent pilots for simulated rotary wing aircraft*. In Proceedings of the Fifth Conference on Computer-Generated Forces and Behavioral Representation, pp. 39-44.
- 22) van Lent, M., & Wray, R. (1994). *A Very Low Cost System for Direct Human Control of Simulated Vehicles*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

ModSAF

1) Vralik, R., & Richardson, W. (1994). *Benchmarking and Optimization of ModSAF*, Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

2) *Functional Description Document for ModSAF*, 29 September 1995.

AFIT Automated Wingman

1) Edwards, M. M. (1995). *The Automated Wingman: A Computer Generated Companion for Users of DIS Compatible Flight Simulators*, Masters Thesis, AFIT.

2) Rumbaugh, J., and others (1991). *Object-Oriented Modeling and Design*. Englewood Cliffs, NJ: Prentice Hall.

3) Cooke, J. M. (1992). NPSNET: *Flight Simulation Dynamic Modeling Using Quaternions*, Presence, 1(4), 404-420.

CFOR

1) Tseng, D. Y., & Howard, M. (February 12, 1995). *Command and Control for Computer-Generated Forces*, a program kick-off briefing, available from Hughes Research Lab.

2) Keirsy, D., Krozel, J., Payton, D., & Tseng, D. (1994). *Case-Based Computer-Generated Forces*. Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

3) Harmon, S. Y., Yang, S. C., & Tseng, D. Y. (1994). *Command and Control Simulation for Computer-Generated Forces*, Proceedings of the Fourth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

4) Howard, M., Hoff, B., & Tseng, D. (1995). *Individual Combatant Development in ModSAF*, Proceedings of the Fifth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

5) Salisbury, M. R., Booker, L. B., Seidel, D. W., & Dahman, J. S. (1995). *ARPA CFOR Briefing*. Proceedings of the Fifth Conference on Computer-Generated Forces and Behavioral Representation. Orlando, FL.

6) *Command and Control Simulation Interface Language (CCSIL) Briefing*, Oct 1995, available from the MITRE Corp.

7) Dahmann, J. S., Salisbury, M. R., Booker, L. B., & Seidel, D. W. (1994). "Command Forces: An Extension of DIS Virtual Simulation." Proceedings of the Eleventh Workshop on Standards for the Interoperability of Defense Simulations, 113-117, Orlando, FL.

8) Salisbury, M. R. (1995). *Command and Control Simulation Interface Language (CCSIL): Status Update*, Proceedings of the Twelfth Workshop on Standards for the Interoperability of Defense Simulations, 639-649, Orlando, FL.

9) Salisbury, M. R., Booker, L. B., Seidel, D. W., & Dahmann, J. S. (1995). Implementation of Command Forces (CFOR) Simulation. Proceedings of the Fifth Conference on Computer-Generated Forces and Behavioral Representation, 423-430, Orlando, FL.

10) Salisbury, M. R., Seidel, D. W., & Booker, L. B. (1995). A Brief Review of the Command Forces (CFOR) Program. Proceedings of the Winter Simulation Conference, Arlington, VA.

11) Seidel, D. W., Salisbury, M. R., Booker, L. B., & Dahmann, J. S. (1995). CFOR Approach To Simulation Scalability. Proceedings of the Electronic Conference on Scalability in Training Simulation, The Society for Computer Simulation, Institute for Operations Research and Management Science.

Suppressor/Force Level Simulation

1) Pope, M., & Miller, D.M. (1995). *A Legacy Model for Tomorrow's Training*. Proceedings of the 17th Interservice/Industry Training Systems and Education Conference, Albuquerque, NM.

2) SUPPRESSOR Release 5.3 User's Guide. Electronic Combat Systems Program Office, ASC/RWX, 2145 Monahan Way, Wright-Patterson AFB, OH 45433-7017.

3) SUPPRESSOR Release 5.3 Version Description Document. Electronic Combat Systems Program Office, ASC/RWX, 2145 Monahan Way, Wright-Patterson AFB, OH 45433-7017.

4) SUPPRESSOR Releases 5.2 and 5.3 Analyst's Manual. Electronic Combat Systems Program Office, ASC/RWX, 2145 Monahan Way, Wright-Patterson AFB, OH 45433-7017.

5) Hanford, C. B., Knight, S., Osgood, M., Pope, M., Sardella, J., & Schwalm, S. (1995). *CELLNET: The Enhancement of Fielded Army Aviation Simulators to Provide DIS Interoperability and a Constructive Simulation Linkage*. 12th DIS Workshop on Standards for the Interoperability of Distributed Simulations, Volume I, Position Papers.

6) Hughes Training Inc., (1992). FLS Product Description.

TECHNICAL PUBLICATION / PRESENTATION CONTROL RECORD

1. TITLE Intelligent agents for computer-generated forces		2. MATERIAL IS FOR <input type="checkbox"/> TECHNICAL REPORT <input checked="" type="checkbox"/> TECHNICAL PAPER <input type="checkbox"/> SPECIAL REPORT <input type="checkbox"/> PUBLICATION IN A JOURNAL <input type="checkbox"/> ORAL PRESENTATION <input type="checkbox"/> PROCEEDINGS <input type="checkbox"/> ABSTRACT <input type="checkbox"/> OTHER (Specify)		
3. AUTHOR(S) (LAST NAME, FIRST NAME, MI, RANK - LEAD AUTHOR FIRST) George, G.R., Mallery, E., & Pope, M.		4. NAME OF JOURNAL OR DETAILS (Date and place) OF ORAL PRESENTATION (indicate if Foreign) Refereed? <input type="checkbox"/> Yes <input type="checkbox"/> No		
5. PROJECT/TASK/WORK UNIT NO. 2743-25-06	6. CONTRACT NO. F41624-95-C-5011	7. PROTOCOL NUMBER		
8. AUTHOR/CONTRACT MONITOR (NAME/OFFICE SYMBOL/EXT) Dr Herb Bell/HRAU/6561		9. CONTRACTOR Hughes Training Inc.		
10. DISTRIBUTION STATEMENT (AFI 61-204) (Select distribution statement from reverse) <input checked="" type="checkbox"/> A* (Public release) <input type="checkbox"/> B <input type="checkbox"/> C <input type="checkbox"/> D <input type="checkbox"/> E <input type="checkbox"/> F <input type="checkbox"/> EXPORT CONTROLLED				
11. SECURITY CLASSIFICATION <input checked="" type="checkbox"/> UNCLASSIFIED <input type="checkbox"/> OTHER (Specify) <input type="checkbox"/> DESTRUCTION NOTICE				
12. RELEASE FOR PUBLICATION REQUIRED FROM ANOTHER AGENCY <input type="checkbox"/> YES <input checked="" type="checkbox"/> NO If yes, attach copy of release				
13. JOINT PUBLICATION WITH ANOTHER GOV'T ORGANIZATION <input type="checkbox"/> YES <input checked="" type="checkbox"/> NO If yes, attach copy of other organization's release				
14. RELEASE FOR USE OF COPYRIGHTED MATERIAL ON MS PAGE IS REQUIRED (attach copy of release)				
15. SPECIAL DISTRIBUTION LIST <input checked="" type="checkbox"/> YES (Include list) <input type="checkbox"/> NO			17. PUBLICATION NO. AL/HR-TP-1996- 0013	
16. LAST PUBLICATION FOR WORK UNIT <input type="checkbox"/> YES <input type="checkbox"/> NO			<input type="checkbox"/> INTERIM <input type="checkbox"/> FINAL	
18. I HAVE REVIEWED THE ATTACHED MATERIALS AND HAVE DETERMINED THEY ARE TECHNICALLY ACCEPTABLE				
OFFICE	SIGNATURE	OFFICE SYMBOL	DATE IN	DATE OUT
HRA STINFO	<i>[Signature]</i>	HRAP/Casey	7-May-96	14-May-96
CONTRACT MONITOR	<i>[Signature]</i>	HRAD/Mudd	14-May-96	14 May 96
TECHNICAL MONITOR				
BRANCH CHIEF	<i>[Signature]</i>	HRAU/Bell	14 May 96	22 May 96
TECHNICAL DIRECTOR	<i>[Signature]</i>	HRA/Martin		28 May 96
DIVISION CHIEF	<i>[Signature]</i>	HRA/Carroll	28 May	28 May 96
HRA STINFO	<i>[Signature]</i>	HRAP/Casey	29	6 May 96
STINFO	<i>[Signature]</i>	Return to author		
CHIEF SCIENTIST	<i>[Signature]</i>	Mudd		
19. EDITING AND FINAL PROCESSING				
DATE IN	TO	DATE OUT	COMMENT	SIGNATURE
	FOREIGN DISCLOSURE			
	EDITING			
	PUBLIC AFFAIRS		CASE FILE NO:	
	COMPOSITION			
	AUTHOR PROOF			
	STINFO			
	FINAL EDIT		PUBLICATION NO:	
	PRINT		NO OF COPIES:	
	DIST (DTIC)		AD NO:	DATE PUBLISHED:
REMARKS: AL/DOKE <input type="checkbox"/> EDIT <input type="checkbox"/> NO EDIT DRAFT CONTRACTOR-PREPARED TECHNICAL PAPER. CONTRACTOR FORMAT IS ACCEPTABLE.				
* I HAVE REVIEWED ATTACHED MATERIAL AND HAVE DETERMINED THAT IT IS UNCLASSIFIED, TECHNICALLY ACCURATE AND SUITABLE FOR RELEASE.				